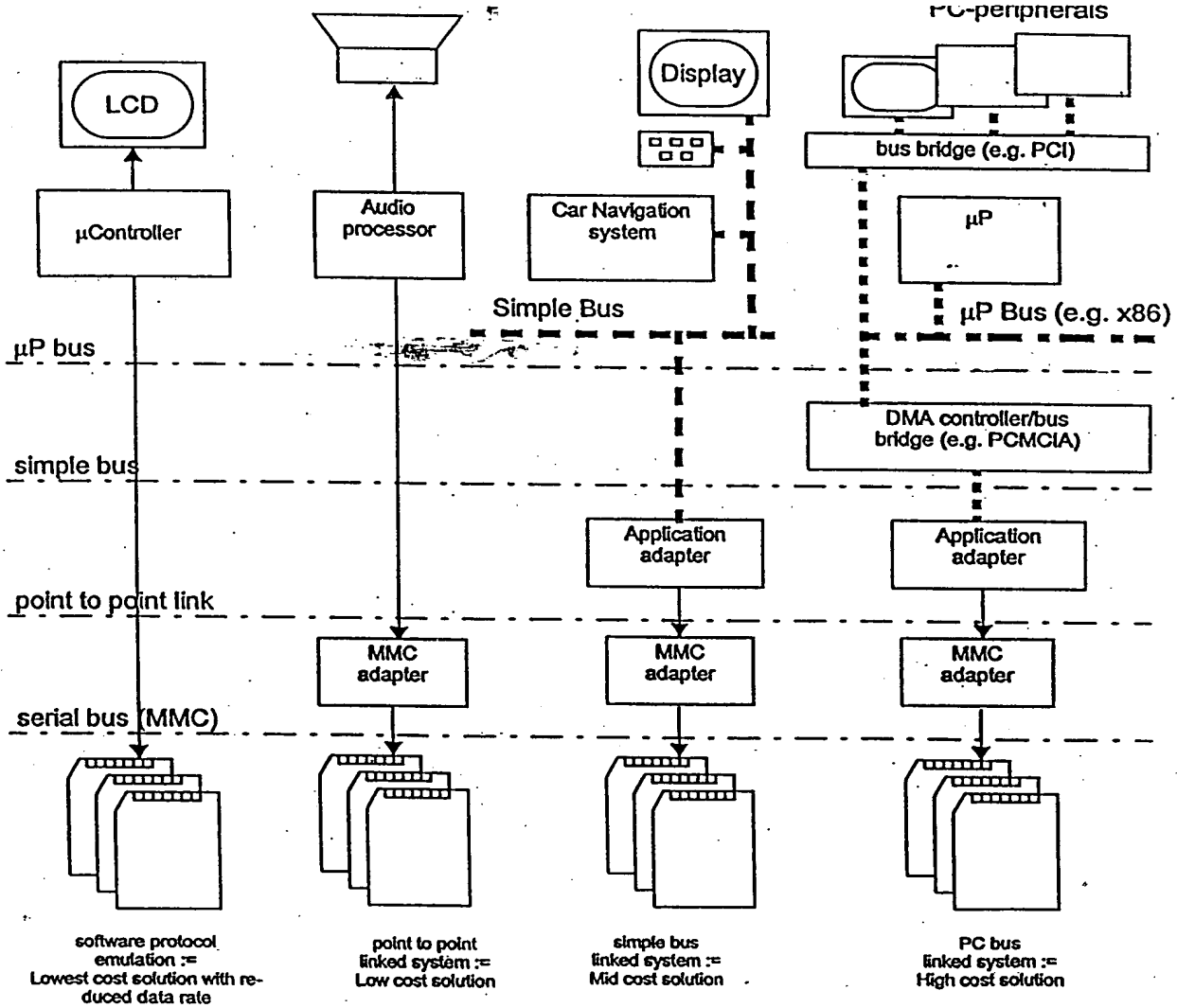



Fig 1

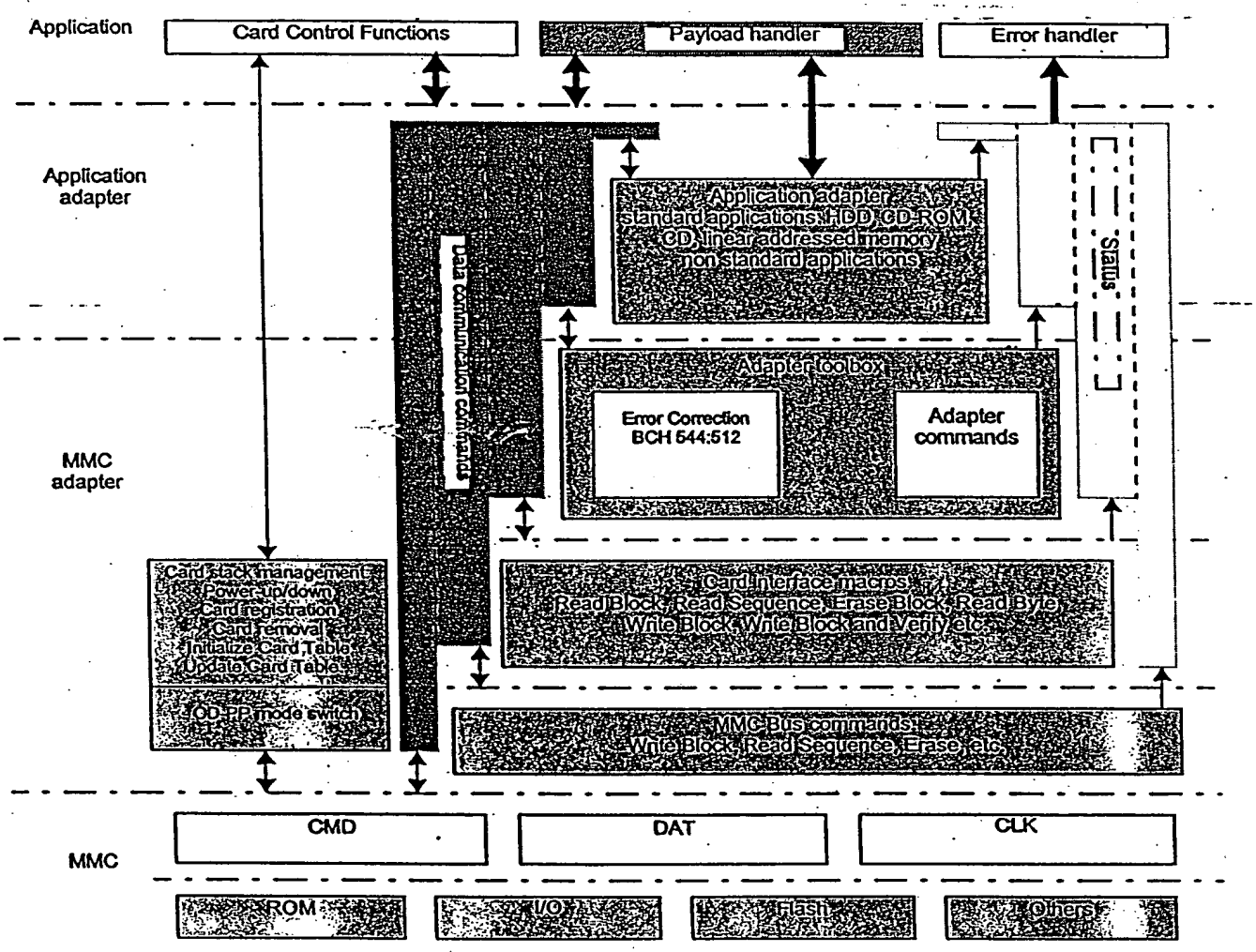

Application
Card Control Functions
Payload handler
Error handler
Application adapter
Application adapter, standard applications, HDD, CD-ROM, CD, linear addressed memory, non standard applications
Status
Data communication commands
Adapter toolbox
Error Correction BCH 544:512
Adapter commands
MMC adapter
Card stack management, Power up/down, Card registration, Card removal, Initialize Card Table, Update Card Table
Card interface macros, Read Block, Read Sequence, Erase Block, Read Byte, Write Block, Write Block and Verify etc.
OD-PP mode switch
MMC Bus commands, Write Block, Read Sequence, Erase, etc.
CMD
DAT
CLK
MMC
ROM
I/O
Flash
Others

Fig 2

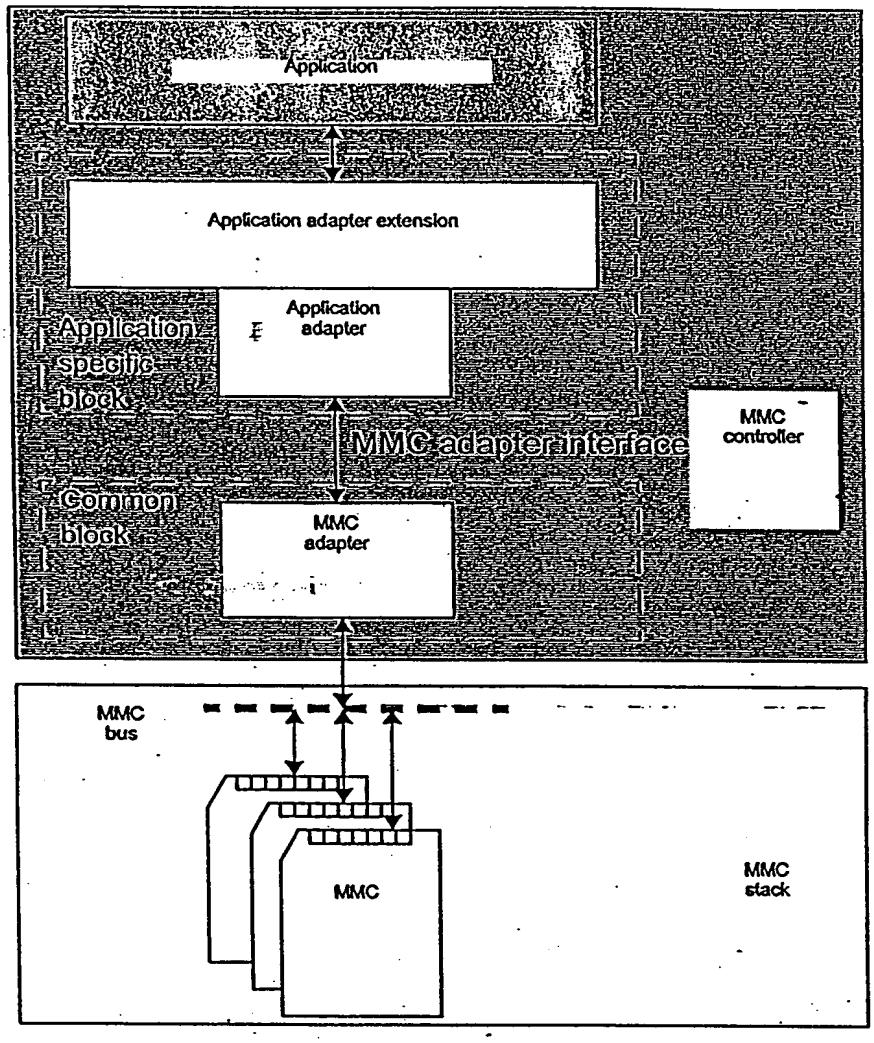

Application
Application adapter extension
Application adapter
Application specific block
MMC adapter interface
MMC controller
Common block
MMC adapter
MMC bus
MMC
MMC stack

Fig 3

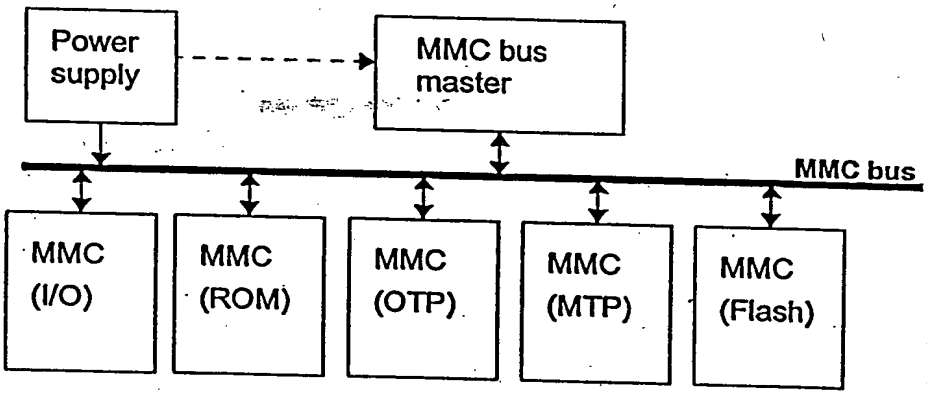

Power supply
MMC bus master
MMC bus
MMC (I/O)
MMC (ROM)
MMC (OTP)
MMC (MTP)
MMC (Flash)

Fig 4

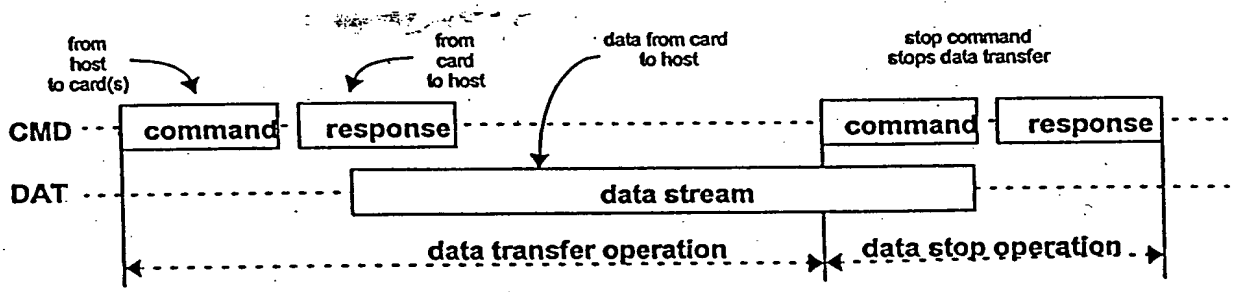

from host to card(s)
from card to host
data from card to host
stop command stops data transfer
CMD
command
response
command
response
DAT
data stream
data transfer operation
data stop operation

Fig 5

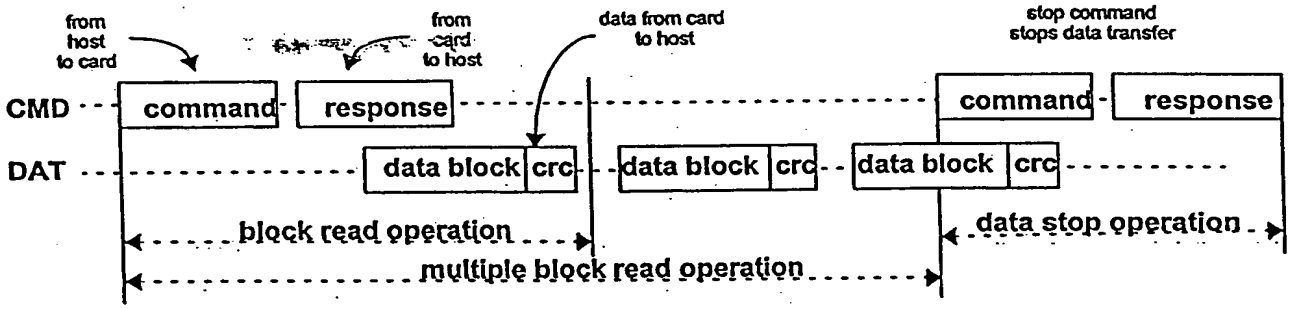

from host to card
from card to host
data from card to host
stop command stops data transfer
CMD
command
response
command
response
DAT
data block
crc
data block
crc
data block
crc
block read operation
data stop operation
multiple block read operation

Fig 6

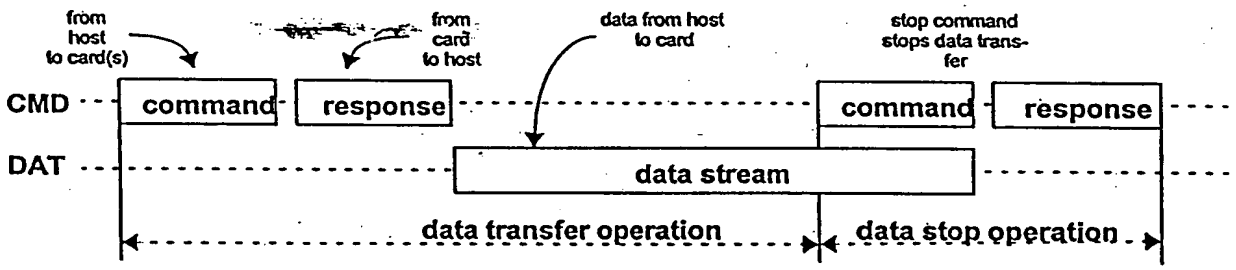

from host to card(s)
from card to host
data from host to card
stop command stops data transfer
CMD
command
response
command
response
DAT
data stream
data transfer operation
data stop operation

Fig 7

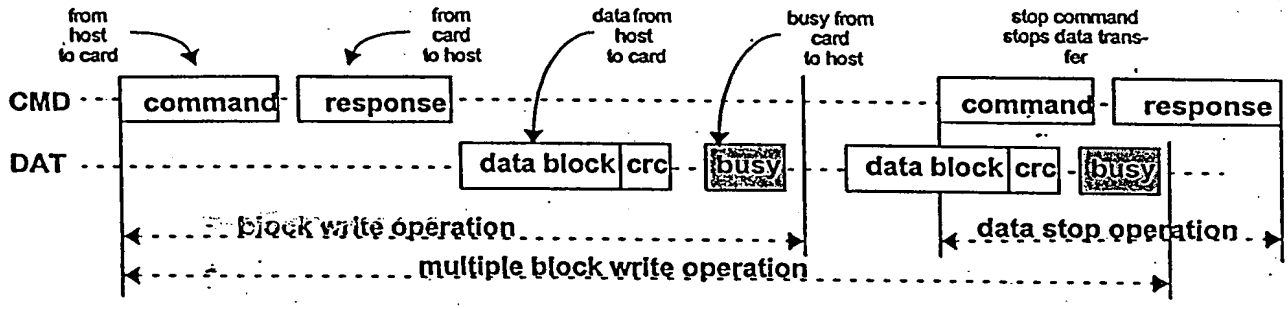

from host to card
from card to host
data from host to card
busy from card to host
stop command stops data transfer
CMD
command
response
command
response
DAT
data block
crc
busy
data block
crc
busy
block write operation
data stop operation
multiple block write operation

Fig 8

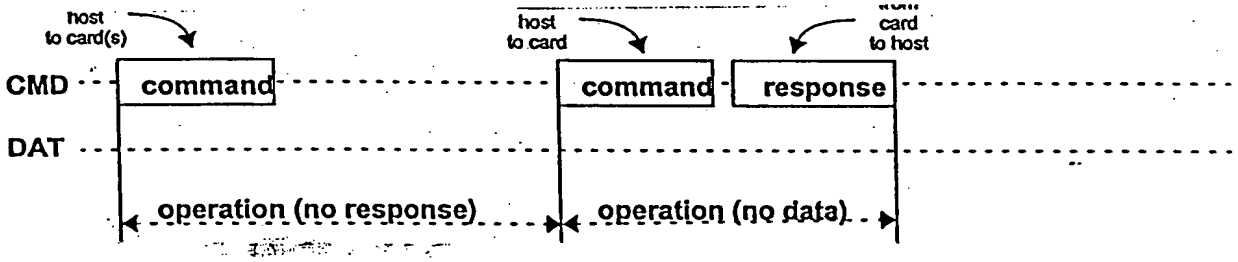

host
to card(s)
host
to card
card
to host
CMD
command
command
response
DAT
operation (no response)
operation (no data)

Fig 9

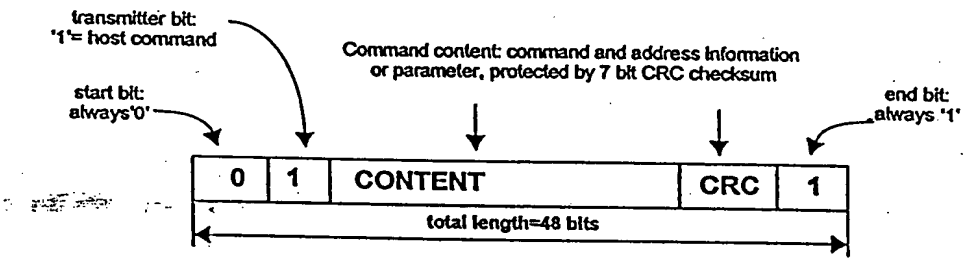

transmitter bit:
'1'= host command
start bit:
always'0'
Command content: command and address information
or parameter, protected by 7 bit CRC checksum
end bit:
always '1'
0
1
CONTENT
CRC
1
total length=48 bits

Fig 10

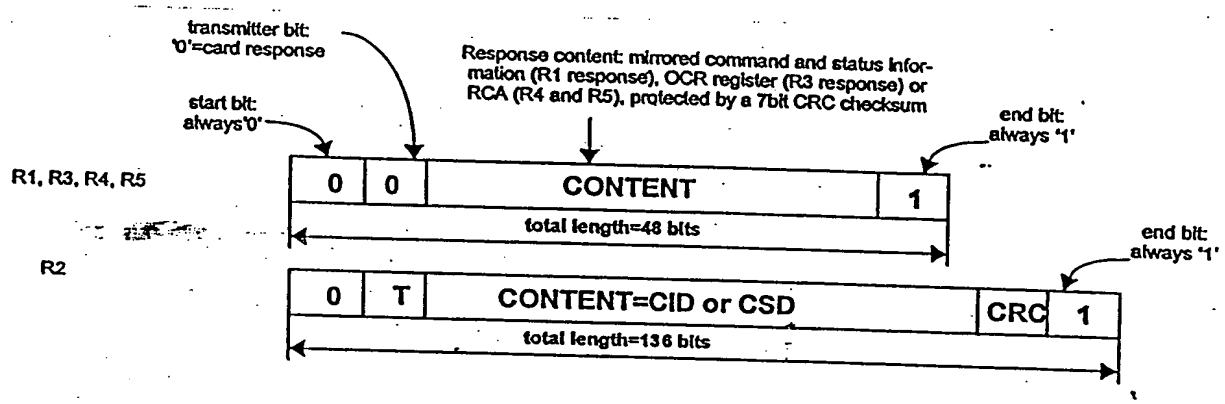

transmitter bit:
'0'=card response
start bit:
always'0'
Response content: mirrored command and status information (R1 response), OCR register (R3 response) or RCA (R4 and R5), protected by a 7bit CRC checksum
end bit:
always '1'
R1, R3, R4, R5
0
0
CONTENT
1
total length=48 bits
end bit:
always '1'
R2
0
T
CONTENT=CID or CSD
CRC
1
total length=136 bits

Fig 11

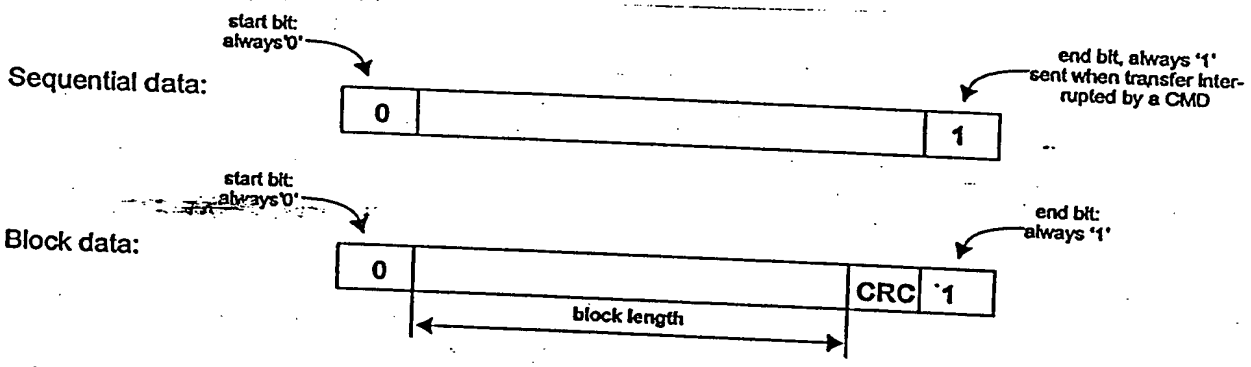

start bit:
always '0'
Sequential data:
0
1
end bit, always '1'
sent when transfer inter-
rupted by a CMD
start bit:
always '0'
Block data:
0
CRC
1
end bit:
always '1'
block length

Fig 12

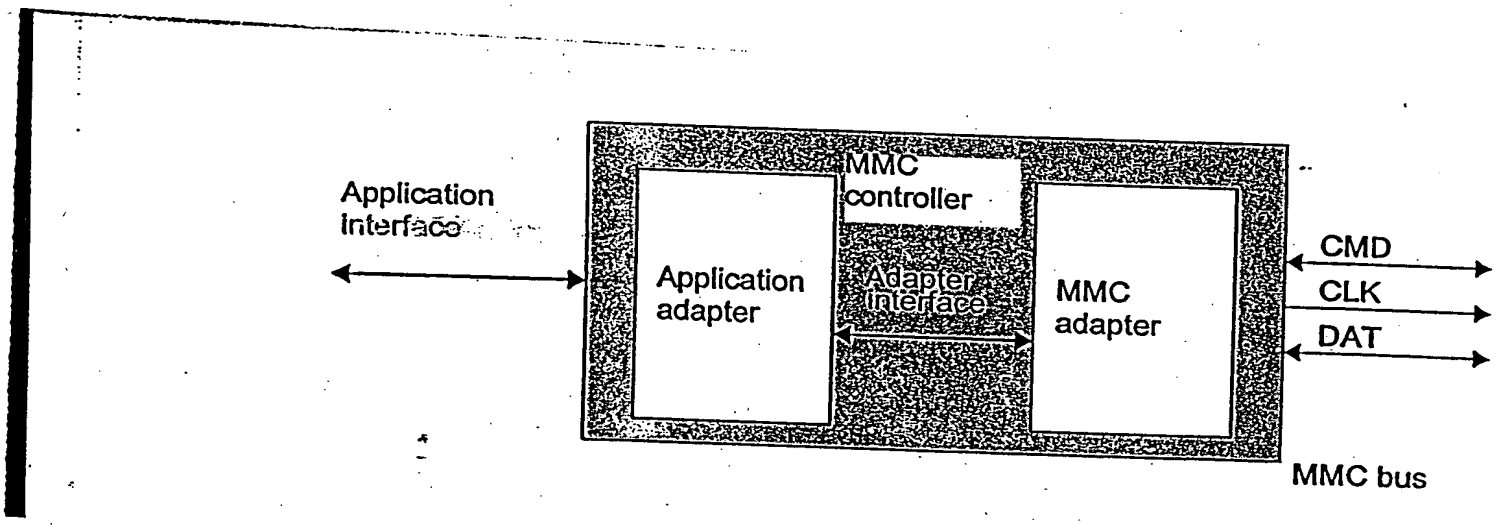

Application interface
MMC controller
Application adapter
Adapter interface
MMC adapter
CMD
CLK
DAT
MMC bus

Fig. 13

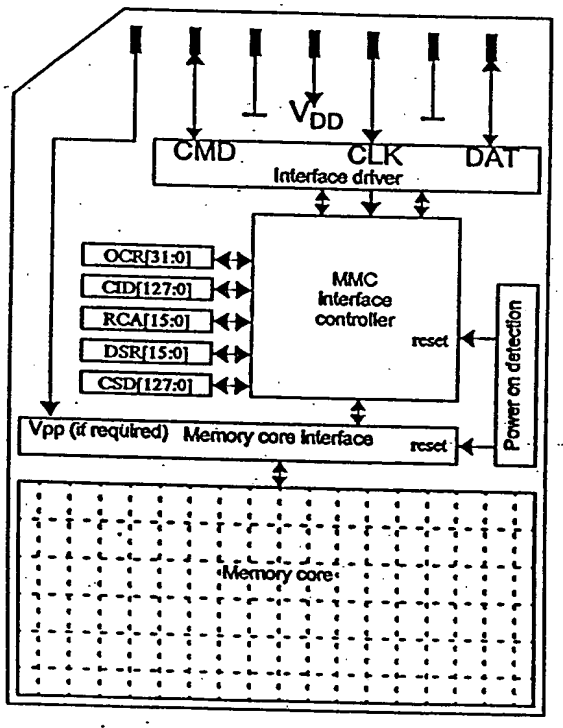

$V_{DD}$
CMD
CLK
DAT
Interface driver
OCR[31:0]
CID[127:0]
RCA[15:0]
DSR[15:0]
CSD[127:0]
MMC
Interface
controller
reset
Power on detection
Vpp (if required)
Memory core interface
reset
Memory core

Fig 14

| OCR bit position | VDD voltage window |
|---|---|
| 0-7 | reserved |
| 8 | 2.0-2.1 |
| 9 | 2.1-2.2 |
| 10 | 2.2-2.3 |
| 11 | 2.3-2.4 |
| 12 | 2.4-2.5 |
| 13 | 2.5-2.6 |
| 14 | 2.6-2.7 |
| 15 | 2.7-2.8 |
| 16 | 2.8-2.9 |
| 17 | 2.9-3.0 |
| 18 | 3.0-3.1 |
| 19 | 3.1-3.2 |
| 20 | 3.2-3.3 |
| 21 | 3.3-3.4 |
| 22 | 3.4-3.5 |
| 23 | 3.5-3.6 |
| 24-30 | reserved |
| 31 | card power up status bit (busy)[1] |

1)This bit is set to LOW if the card has not finished the power up routine

Fig 15

| Name | Field | Width | CID-slice |
|---|---|---|---|
| Manufacturer ID | MID | 24 | [127:104] |
| Card Individual number | CIN | 96 | [103:8] |
| CRC7 checksum | CRC | 7 | [7:1] |
| not used, always '1' | - | 1 | [0:0] |

Fig 16

| Name | Field | Width | Cell Type | CSD-slice |
|---|---|---|---|---|
| CSD structure | CSD_STRUCTURE | 2 | R | [127:126] |
| MMC protocol version | MMC_PROT | 4 | R | [125:122] |
| reserved | | 2 | R | [121:120] |
| data read access-time-1 | TAAC | 8 | R | [119:112] |
| data read access-time-2 in CLK cycles (NSAC*100) | NSAC | 8 | R | [111:104] |
| max. data transfer rate | TRAN_SPEED | 8 | R | [103:96] |
| card command classes | CCC | 12 | R | [95:84] |
| max. read data block length | READ_BL_LEN | 4 | R | [83:80] |
| partial blocks for read allowed | READ_BL_PARTIAL | 1 | R | [79:79] |
| write block misalignment | WRITE_BLK_MISALIGN | 1 | R | [78:78] |
| read block misalignment | READ_BLK_MISALIGN | 1 | R | [77:77] |
| DSR implemented | DSR_IMP | 1 | R | [76:76] |
| external Vpp | VPROG | 2 | R | [75:74] |
| device size mantissa | C_SIZE_MANT | 8 | R | [73:66] |
| device size exponent | C_SIZE_EXP | 4 | R | [65:62] |
| max. read current @$V_{DD}$ min | VDD_R_CURR_MIN | 3 | R | [61:59] |
| max. read current @$V_{DD}$ max | VDD_R_CURR_MAX | 3 | R | [58:56] |

| Name | Field | Width | Cell Type | CSD-slice |
|---|---|---|---|---|
| max. write current @$V_{DD}$ min | VDD_W_CURR_MIN | 3 | R | [55:53] |
| max. write current @$V_{DD}$ max | VDD_W_CURR_MAX | 3 | R | [52:50] |
| max. $V_{PP}$ current | VPP_CURR | 3 | R | [49:47] |
| erase sector size | SECTOR_SIZE | 5 | R | [46:42] |
| erase group size | ERASE_GRP_SIZE | 5 | R | [41:37] |
| write protect group size | WP_GRP_SIZE | 5 | R | [36:32] |
| write protect group enable | WP_GRP_ENABLE | 1 | R | [31:31] |
| manufacturer default ECC | DEFAULT_ECC | 2 | R | [30:29] |
| stream write speed factor | R2W_FACTOR | 3 | R | [28:26] |
| max. write data block length | WRITE_BL_LEN | 4 | R | [25:22] |
| partial blocks for write allowed | WRITE_BL_PARTIAL | 1 | R | [21:21] |
| reserved | | 5 | R | [20:16] |
| reserved | | 3 | R/W | [15:13] |
| copy flag (OTP) | COPY | 1 | R/W | [12:12] |
| permanent write protection | PERM_WRITE_PROTECT | 1 | R/W | [11:11] |
| temporary write protection | TMP_WRITE_PROTECT | 1 | R/W/E | [10:10] |
| ECC code | ECC | 2 | R/W/E | [9:8] |
| CRC | CRC | 7 | R/W/E | [7:1] |
| not used, always '1' | - | 1 | - | [0:0] |

Fig 17

| CSD_STRUCTURE | CSD structure version | Valid for MMC protocol version |
|---|---|---|
| 0 | CSD version No. 1.0 | MMC protocol version 1.0-1.2 |
| 1 | CSD version No. 1.1 | MMC protocol version 1.4 |
| 2-3 | reserved | |

Fig 18

| MMC_PROT | MMC protocol version |
|---|---|
| 0 | MMC protocol version 1.0-1-2 |
| 1 | MMC protocol version 1.3 |
| 2-15 | reserved |

Fig 19

| TAAC bit position | code |
|---|---|
| 2:0 | time exponent<br>0=1ns, 1=10ns, 2=100ns, 3=1μμs, 4=10μμs, 5=100μμs, 6=1ms, 7=10ms |
| 6:3 | time mantissa<br>0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5, 5=2.0, 6=2.5, 7=3.0, 8=3.5, 9=4.0, A=4.5, B=5.0, C=5.5, D=6.0, E=7.0, F=8.0 |
| 7 | reserved |

Fig 20

| TRAN_SPEED bit | code |
|---|---|
| 2:0 | transfer rate exponent<br>0=100kbit/s, 1=1Mbit/s, 2=10Mbit/s,<br>3=100Mbit/s, 4... 7=reserved |
| 6:3 | time mantissa<br>0=reserved, 1=1.0, 2=1.2, 3=1.3, 4=1.5,<br>5=2.0, 6=2.5, 7=3.0, 8=3.5, 9=4.0, A=4.5,<br>B=5.0, C=5.5, D=6.0, E=7.0, F=8.0 |
| 7 | reserved |

Fig 21

| CCC bit | Supported card command class |
|---|---|
| 0 | class 0 |
| 1 | class 1 |
| ..... | |
| 11 | class 11 |

Fig 22

| BL_LEN | Block length | Remark |
|---|---|---|
| 0 | $2^0$ = 1 Byte | |
| 1 | $2^1$ = 2 Bytes | |
| ..... | | |
| 11 | $2^{11}$ = 2048 Bytes | |
| 12-14 | reserved | |
| 15 | any | can be set by the host in 1 Byte steps between 1 Byte and (theoretically) the total device size |

Fig 23

| DSR_IMP | DSR type |
|---|---|
| 0 | no DSR implemented |
| 1 | DSR implemented |

Fig 24

| VDD_R_CURR_MIN<br>VDD_W_CURR_MIN | code for current consumption @ $V_{DD}$ |
|---|---|
| 2:0 | 0=0.5mA; 1=1mA; 2=5mA; 3=10mA; 4=25mA;<br>5=35mA; 6=60mA; 7=100mA |

| VDD_R_CURR_MAX<br>VDD_W_CURR_MAX | code for current consumption @ $V_{DD}$ |
|---|---|
| 2:0 | 0=1mA; 1=5mA; 2=10mA; 3=25mA; 4=35mA;<br>5=45mA; 6=80mA; 7=200mA |

Fig 25

| R2W_FACTOR | Multiples of read access time |
|---|---|
| 0 | 1 |
| 1 | 2 (write half as fast as read) |
| 2 | 4 |
| 3 | 8 |
| 4 | 16 |
| 5 | 32 |
| 6,7 | reserved |

Fig 26

| ECC | ECC type | Maximum number of correctable bits per block |
|---|---|---|
| 0 | none (default) | none |
| 1 | BCH (542,512) | 3 |
| 2-3 | reserved | - |

Fig 27

| CSD Field | Command classes | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| CSD_STRUCTURE | + | + | + | + | + | + | + | + | + | + |
| MMC_PROT | + | + | + | + | + | + | + | + | + | + |
| TAAC | | + | + | + | + | + | + | + | + | |
| NSAC | | + | + | + | + | + | + | + | + | |
| TRAN_SPEED | | + | + | + | + | | | | | |
| CCC | + | + | + | + | + | + | + | + | + | + |
| READ_BL_LEN | | | + | | | | | | | |
| READ_BL_PARTIAL | | | + | | | | | | | |
| WRITE_BLK_MISALIGN | | | | | + | | | | | |
| READ_BLK_MISALIGN | | | + | | | | | | | |
| DSR_IMP | + | + | + | + | + | + | + | + | + | + |
| VPROG | | | | + | + | + | + | + | + | |
| C_SIZE_MANT | | + | + | + | + | + | + | + | + | |
| C_SIZE_EXP | | + | + | + | + | + | + | + | + | |
| VDD_R_CURR_MIN | | + | + | | | | | | | |
| VDD_R_CURR_MAX | | + | + | | | | | | | |
| VDD_W_CURR_MIN | | | | + | + | + | + | + | + | |
| VDD_W_CURR_MAX | | | | + | + | + | + | + | + | |
| VPP_CURR | | | | + | + | + | + | + | + | |
| SECTOR_SIZE | | | | | | + | + | + | + | |
| ERASE_GRP_SIZE | | | | | | + | + | + | + | |
| WP_GRP_SIZE | | | | | | | + | + | + | |
| WP_GRP_ENABLE | | | | | | | + | + | + | |
| DEFAULT_ECC | | + | + | + | + | + | + | + | + | |
| R2W_FACTOR | | | | + | + | + | + | + | + | |
| WRITE_BL_LEN | | | | + | + | + | + | + | + | |
| WRITE_BL_PARTIAL | | | | + | + | + | + | + | + | |
| COPY | + | + | + | + | + | + | + | + | + | + |
| PERM_WRITE_PROTECT | + | + | + | + | + | + | + | + | + | + |
| TMP_WRITE_PROTECT | + | + | + | + | + | + | + | + | + | + |
| ECC | | + | + | + | + | + | + | + | + | |
| CRC | + | + | + | + | + | + | + | + | + | + |

Fig 28

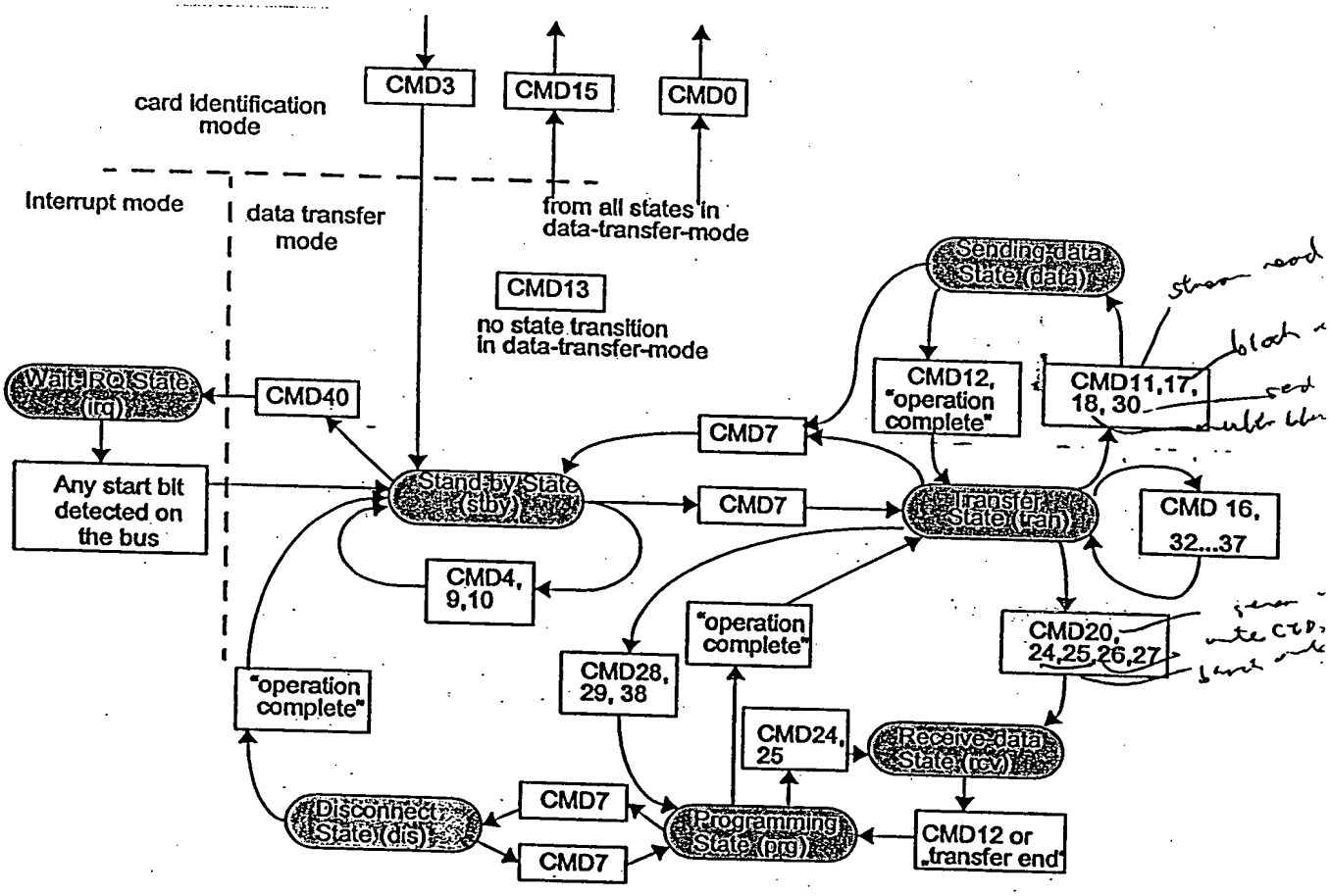

card identification mode
CMD3
CMD15
CMD0
Interrupt mode
data transfer mode
from all states in data-transfer-mode
CMD13
no state transition in data-transfer-mode
Sending-data State (data)
CMD12, "operation complete"
CMD11,17, 18, 30
Wait-IRQ State (irq)
CMD40
CMD7
Any start bit detected on the bus
Stand-by State (stby)
CMD7
Transfer State (tran)
CMD 16, 32...37
CMD4, 9,10
"operation complete"
CMD20, 24,25,26,27
CMD28, 29, 38
"operation complete"
CMD24, 25
Receive-data State (rcv)
Disconnect State (dis)
CMD7
Programming State (prg)
CMD12 or transfer end
CMD7

Fig 29

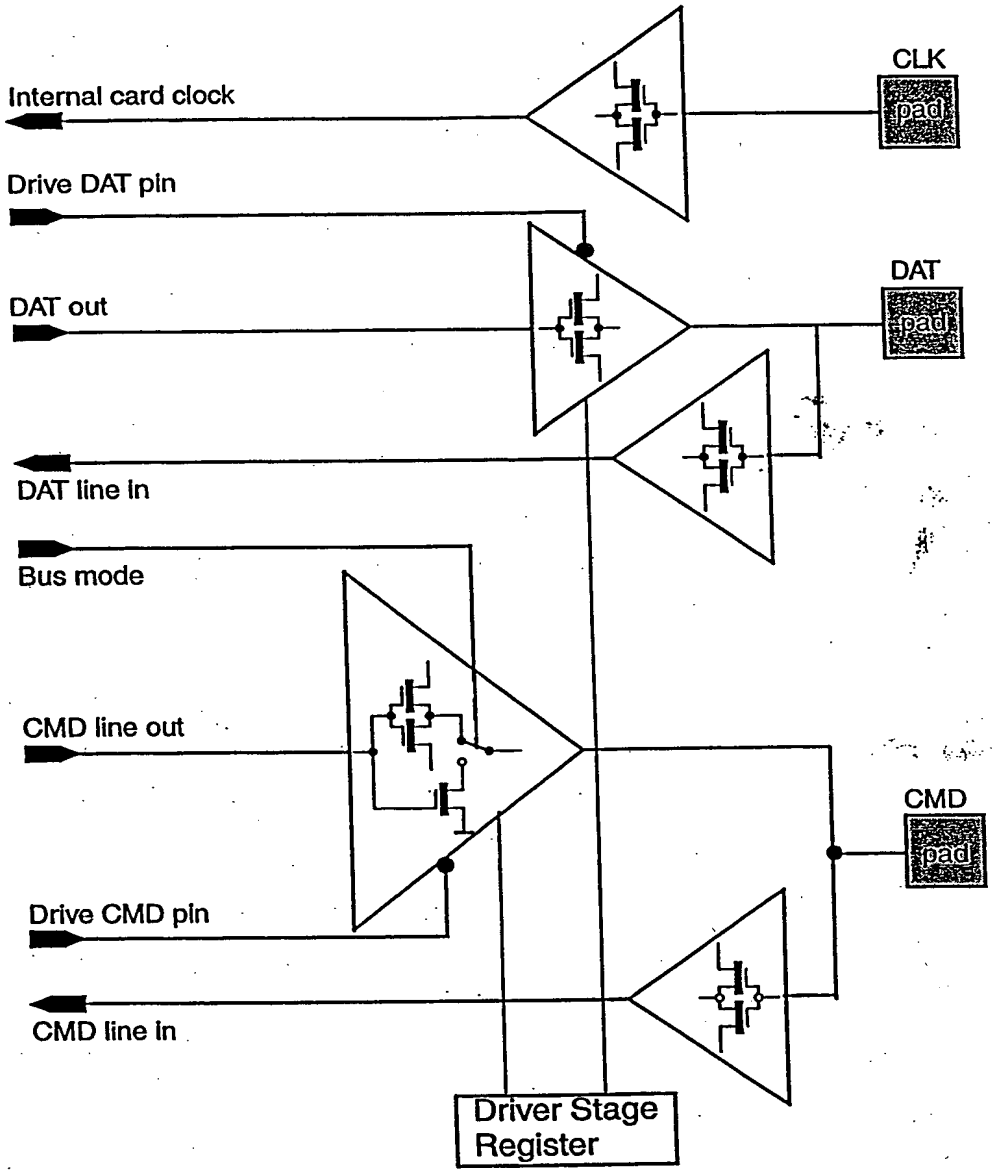

Internal card clock
CLK
pad
Drive DAT pin
DAT out
DAT
pad
DAT line in
Bus mode
CMD line out
CMD
pad
Drive CMD pin
CMD line in
Driver Stage Register

Fig 30

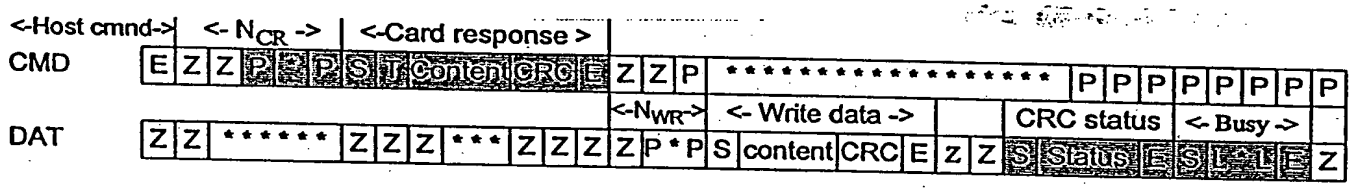

<Host cmnd->
<- $N_{CR}$ ->
<Card response >
CMD
Content
CRC
<-$N_{WR}$->
<- Write data ->
CRC status
<- Busy ->
DAT
content
CRC
Status

Fig 31

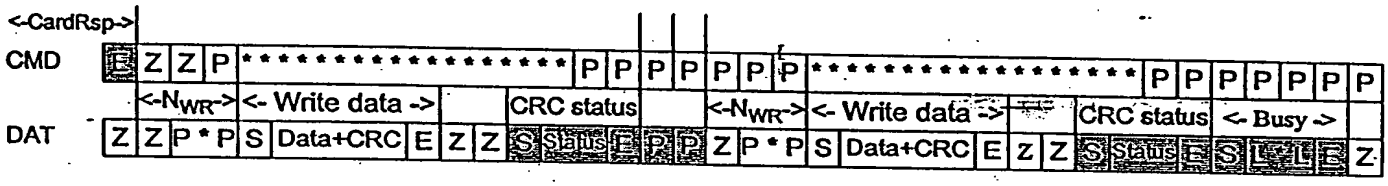

<-CardRsp->
CMD
DAT
<-N_WR->
<- Write data ->
CRC status
<-N_WR->
<- Write data ->
CRC status
<- Busy ->
Data+CRC
Data+CRC

Fig 32

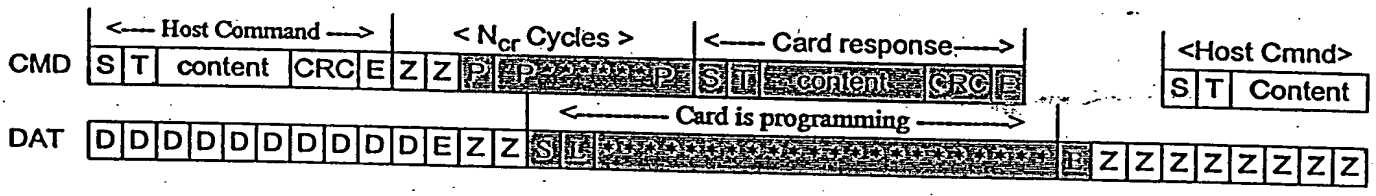
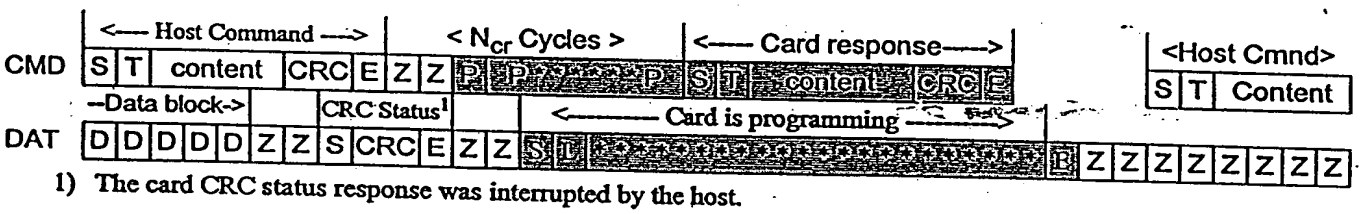

Host Command
< $N_{cr}$ Cycles >
Card response
<Host Cmnd>
CMD
S T content CRC E Z Z P P P S T content CRC E
S T Content
Card is programming
DAT
D D D D D D D D D D E Z Z S E E Z Z Z Z Z Z Z Z

Fig 33




1) The card CRC status response was interrupted by the host.

Fig 34

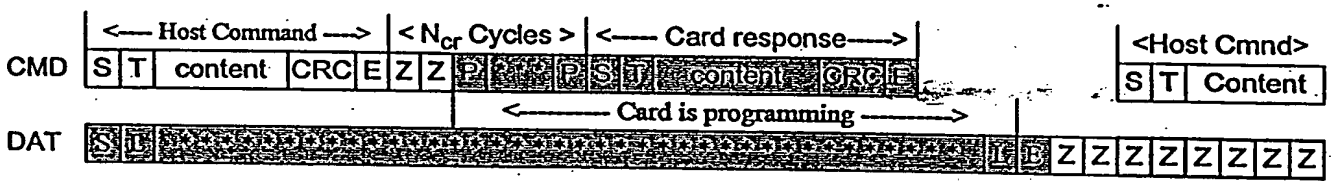

Host Command
< Ncr Cycles >
Card response
<Host Cmnd>
CMD
S T content CRC E Z Z P P P P S T content CRC E
S T Content
Card is programming
DAT
S B Z Z Z Z Z Z Z Z

Fig 35

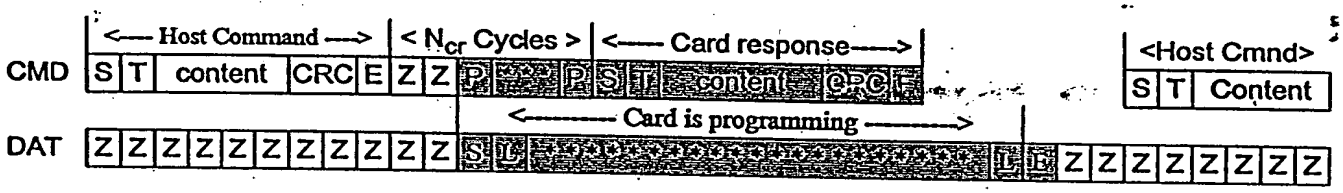

Host Command
< Ncr Cycles >
Card response
<Host Cmnd>
CMD
S T content CRC E Z Z
P S T content CRC E
S T Content
Card is programming
DAT
Z Z Z Z Z Z Z Z Z Z
S
E
Z Z Z Z Z Z Z Z

Fig 36

| Card Command Class (CCC) | class description | Supported commands | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 7 | 9 | 10 | 11 | 12 | 13 | 15 | 16 | 17 | 18 | 20 |
| class 0 | basic | + | + | + | + | + | + | + | + | | + | + | + | | | | |
| class 1 | stream read | | | | | | | | | + | | | | | | | |
| class 2 | block read | | | | | | | | | | | | | + | + | + | |
| class 3 | stream write | | | | | | | | | | | | | | | | + |
| class 4 | block write | | | | | | | | | | | | | + | | | |
| class 5 | erase | | | | | | | | | | | | | | | | |
| class 6 | write write-protection | | | | | | | | | | | | | | | | |
| class 7 | read write-protection | | | | | | | | | | | | | | | | |
| class 8 | erase write-protection | | | | | | | | | | | | | | | | |
| class 9 | I/O mode | | | | | | | | | | | | | | | | |
| class 10-11 | reserved | | | | | | | | | | | | | | | | |

| Card Command Class (CCC) | class description | Supported commands | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | | |
| class 0 | basic | | | | | | | | | | | | | | | | | | |
| class 1 | stream read | | | | | | | | | | | | | | | | | | |
| class 2 | block read | | | | | | | | | | | | | | | | | | |
| class 3 | stream write | | | | | | | | | | | | | | | | | | |
| class 4 | block write | + | + | + | + | | | | | | | | | | | | | | |
| class 5 | erase | | | | | | | | + | + | + | + | + | + | + | | | | |
| class 6 | write write-protection | | | | | + | | + | | | | | | | | | | | |
| class 7 | read write-protection | | | | | | | + | | | | | | | | | | | |
| class 8 | erase write-protection | | | | | + | + | + | | | | | | | | | | | |

| Card Command Class (CCC) | class description | Supported commands | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | | |
| class 9 | I/O mode | | | | | | | | | | | | | | | + | + | | |
| class 10-11 | reserved | | | | | | | | | | | | | | | | | | |

Fig 37

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD0 | bc | [31:0] stuff bits | - | GO_IDLE_STATE | resets all cards to idle state |
| CMD1 | bcr | [31:0] OCR without busy | R3 | SEND_OP_COND | asks all cards in idle state to send their operation conditions register contents in the response on the CMD line. |
| CMD2 | bcr | [31:0] stuff bits | R2 | ALL_SEND_CID | asks all cards to send their CID numbers on the CMD line |
| CMD3 | ac | [31:16] RCA [15:0] stuff bits | R1 | SET_RELATIVE_ ADDR | assigns relative address to the card |
| CMD4 | bc | [31:16] DSR [15:0] stuff bits | - | SET_DSR | programs the DSR of all cards |
| CMD5 | reserved | | | | |
| CMD6 | reserved | | | | |
| CMD7 | ac | [31:16] RCA [15:0] stuff bits | R1 (only from the selected card) | SELECT/ DESELECT_ CARD | command toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases the card is selected by its own relative address and gets deselected by any other address; address 0 deselects all. |
| CMD8 | reserved | | | | |
| CMD9 | ac | [31:16] RCA [15:0] stuff bits | R2 | SEND_CSD | addressed card sends its card-specific data (CSD) on the CMD line. |
| CMD10 | ac | [31:16] RCA [15:0] stuff bits | R2 | SEND_CID | addressed card sends its card identification (CID) on CMD the line. |
| CMD11 | adtc | [31:0] data address[1] | R1 | READ_DAT_UNTIL_ STOP | reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows. |
| CMD12 | ac | [31:0] stuff bits | R1 | STOP_ TRANSMISSION | forces the card to stop transmission |

Fig 38

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD13 | ac | [31:16] RCA<br>[15:0] stuff bits | R1 | SEND_STATUS | addressed card sends its status register. |
| CMD14 | reserved | | | | |
| CMD15 | ac | [31:16] RCA<br>[15:0] stuff bits | - | GO_INACTIVE_<br>STATE | sets the card to inactive state in order to protect the card stack against communication breakdowns. |

Fig 38 (cont)

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD16 | ac | [31:0] block length | R1 | SET_BLOCKLEN | sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD. |
| CMD17 | adtc | [31:0] data address | R1 | READ_SINGLE_<br>BLOCK | reads a block of the size selected by the SET_BLOCKLEN command.[1] |
| CMD18 | adtc | [31:0] data address | R1 | READ_MULTIPLE_<br>BLOCK | continuously transfers data blocks from card to host until interrupted by a stop command. |
| CMD19 | reserved | | | | |

Fig 39

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD20 | adtc | [31:0] data address | R1b | WRITE_DAT_UNTI<br>L_<br>STOP | writes data stream from the host, starting at the given address, until a STOP_TRANSMISSION follows. |
| CMD21<br>CMD23 | reserved | | | | |

Fig 40

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD24 | adtc | [31:0] data address | R1b | WRITE_BLOCK | writes a block of the size selected by the SET_BLOCKLEN command.[1] |
| CMD25 | adtc | [31:0] data address | R1b | WRITE_MULTIPL<br>E_<br>BLOCK | continuously writes blocks of data until a STOP_TRANSMISSION follows. |

Fig 41

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD26 | adtc | [31:0] stuff bits | R1b | PROGRAM_CID | programming of the card identification register. This command shall be issued only once per MMC card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer. |
| CMD27 | adtc | [31:0] stuff bits | R1b | PROGRAM_CSD | programming of the programmable bits of the CSD. |

Fig 41 (cont.)

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD28 | ac | [31:0] data address | R1b | SET_WRITE_PROT | If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE). |
| CMD29 | ac | [31:0] data address | R1b | CLR_WRITE_PROT | If the card provides write protection features, this command clears the write protection bit of the addressed group. |
| CMD30 | adtc | [31:0] write protect data address | R1 | SEND_WRITE_PROT | If the card provides write protection features, this command asks the card to send the status of the write protection bits. [1] |
| CMD31 | reserved | | | | |

Fig 42

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD32 | ac | [31:0] data address | R1 | TAG_SECTOR_START | sets the address of the first sector of the erase group. |
| CMD33 | ac | [31:0] data address | R1 | TAG_SECTOR_END | sets the address of the last sector in a continuous range within the selected erase group, or the address of a single sector to be selected for erase. |

Fig 43

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD34 | ac | [31:0] data address | R1 | UNTAG_SECTOR | removes one previously selected sector from the erase selection. |
| CMD35 | ac | [31:0] data address | R1 | TAG_ERASE_GROUP_START | sets the address of the first erase group within a range to be selected for erase |
| CMD36 | ac | [31:0] data address | R1 | TAG_ERASE_GROUP_END | sets the address of the last erase group within a continuous range to be selected for erase |
| CMD37 | ac | [31:0] data address | R1 | UNTAG_ERASE_GROUP | removes one previously selected erase group from the erase selection |
| CMD38 | ac | [31:0] stuff bits | R1b | ERASE | erases all previously selected sectors |

Fig 43 (cont)

| CMD INDEX | type | argument | resp | abbreviation | command description |
|---|---|---|---|---|---|
| CMD39 | ac | [31:16] RCA<br>[15:8] register address<br>[7:0] register data | R4 | FAST_IO | used to write and read 8 bit (register) data fields. The command addresses a card and a register and provides the data. The R4 response contains data read from the addressed register. This command accesses application dependent registers which are not defined in the MMC standard. |
| CMD40 | bcr | [31:0] stuff bits | R5 | GO_IRQ_STATE | Sets the system into interrupt mode |
| CMD41-CMD59 | reserved | | | | |
| CMD60-63 | reserved for manufacturer | | | | |

Fig 44

| Bit position | 47 | 46 | [45:40] | [39:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|
| Width (bits) | 1 | 1 | 6 | 32 | 7 | 1 |
| Value | '0' | '0' | x | x | x | '1' |
| Description | start bit | transmission bit | command index | card status | CRC7 | end bit |

Fig 45

| Bit position | 135 | 134 | [133:128] | [127:1] | 0 |
|---|---|---|---|---|---|
| Width (bits) | 1 | 1 | 6 | 127 | 1 |
| Value | '0' | '0' | '111111' | x | '1' |
| Description | start bit | transmission bit | reserved | CID or CSD register incl. internal CRC7 | end bit |

Fig 46

| Bit position | 47 | 46 | [45:40] | [39:8] | [7:1] | 0 |
|---|---|---|---|---|---|---|
| Width (bits) | 1 | 1 | 6 | 32 | 7 | 1 |
| Value | '0' | '0' | '111111' | x | '1111111' | '1' |
| Description | start bit | transmission bit | reserved | OCR register | reserved | end bit |

Fig 47

| Bit position | 47 | 46 | [45:40] | [39:8] Argument field | | | [7:1] | 0 |
|---|---|---|---|---|---|---|---|---|
| Width (bits) | 1 | 1 | 6 | 16 | 8 | 8 | 7 | 1 |
| Value | '0' | '0' | '100111' | x | x | x | x | '1' |
| Description | start bit | trans-mission bit | CMD39 | RCA [31:16] | register addr. [15:8] | read register contents [7:0] | CRC7 | end bit |

Fig 48

| Bit position | 47 | 46 | [45:40] | [39:8] Argument field | | [7:1] | 0 |
|---|---|---|---|---|---|---|---|
| Width (bits) | 1 | 1 | 6 | 16 | 16 | 7 | 1 |
| Value | '0' | '0' | '101000' | x | x | x | '1' |
| Description | start bit | trans-mission bit | CMD40 | RCA [31:16] of winning card or of the host | [15:0] Not defined. May be used for IRQ data | CRC7 | end bit |

Fig 49

| Bits | Identifier | Type | Value | Description | Clear Condition |
|---|---|---|---|---|---|
| 31 | OUT_OF_RANGE | E R | '0'= no error<br>'1'= error | The command's argument was out of the allowed range for this card. | C |
| 30 | ADDRESS_ERROR | E R X | '0'= no error<br>'1'= error | A misaligned address which did not match the block length was used in the command. | C |
| 29 | BLOCK_LEN_ERROR | E R | '0'= no error<br>'1'= error | The transferred block length is not allowed for this card, or the number of transferred bytes does not match the block length. | C |
| 28 | ERASE_SEQ_ERROR | E R | '0'= no error<br>'1'= error | An error in the sequence of erase commands occurred. | C |
| 27 | ERASE_PARAM | E X | '0'= no error<br>'1'= error | An invalid selection of sectors or groups for erase occurred. | C |
| 26 | WP_VIOLATION | E R X | '0'= not protected<br>'1'= protected | Attempt to program a write protected block. | C |
| 25:24 | reserved | | | | |
| 23 | COM_CRC_ERROR | E R | '0'= no error<br>'1'= error | The CRC check of the previous command failed. | B |
| 22 | ILLEGAL_COMMAND | E R | '0'= no error<br>'1'= error | Command not legal for the card state | B |
| 21 | CARD_ECC_FAILED | E X | '0'= success<br>'1'= failure | Card internal ECC was applied but failed to correct the data. | C |
| 20 | CC_ERROR | E R X | '0'= no error<br>'1'= error | Internal card controller error | C |
| 19 | ERROR | E R X | '0'= no error<br>'1'= error | A general or an unknown error occurred during the operation. | C |
| 18 | UNDERRUN | E X | '0'= no error<br>'1'= error | The card could not sustain data transfer in stream read mode | C |
| 17 | OVERRUN | E X | '0'= no error<br>'1'= error | The card could not sustain data programming in stream write mode | C |
| 16 | CID/ CSD_OVERWRITE | E R | '0'= no error<br>'1'= error | can be either one of the following errors:<br>- The CID register has been already written and can not be overwritten<br>- The read only section of the CSD does not match the card content.<br>- An attempt to reverse the copy (set as original) or permanent WP (unprotected) bits was made. | C |

Fig 50

| Bits | Identifier | Type | Value | Description | Clear Condition |
|---|---|---|---|---|---|
| 15 | WP_ERASE_SKIP | S X | '0'= not protected<br>'1'= protected | Only partial address space was erased due to existing write protected blocks. | C |
| 14 | CARD_ECC_DISABLED | S X | '0'= enabled<br>'1'= disabled | The command has been executed without using the internal ECC. | A |
| 13 | ERASE_RESET | S R | '0'= cleared<br>'1'= set | An erase sequence was cleared before executing because an out of erase sequence command was received | C |
| 12:9 | CURRENT_STATE | S X | 0 = idle<br>1 = ready<br>2 = ident<br>3 = stby<br>4 = tran<br>5 = data<br>6 = rcv<br>7 = prg<br>8 = dis<br>9-15 = reserved | State of the card. The four bits are interpreted as a binary coded number between 0 and 15. | B |
| 8 | READY_FOR_DATA | S X | '0'= not ready<br>'1'= ready | corresponds to buffer empty signalling on the bus | A |
| 7:0 | reserved | | | | |

Fig 50 (cont)

| | |
|---|---|
| S | Start bit (= '0') |
| T | Transmitter bit (Host = '1', Card = '0') |
| P | One-cycle pull-up (= '1') |
| E | End bit (=1) |
| Z | high impedance state (-> = '1') |
| D | Data bits |
| * | repetition |
| CRC | Cyclic redundancy check bits (7 bits) |
| | Card active |
| | Host active |

Fig 51

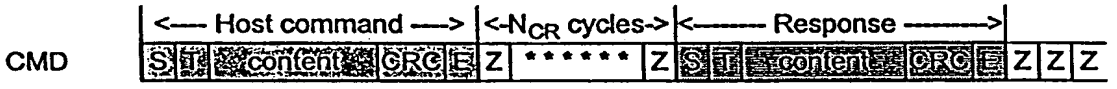



Fig 52

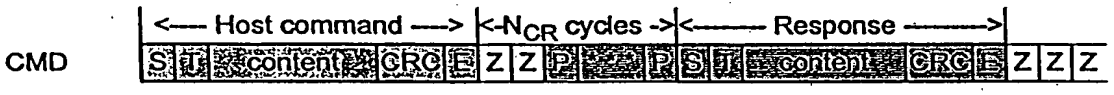



Fig 53

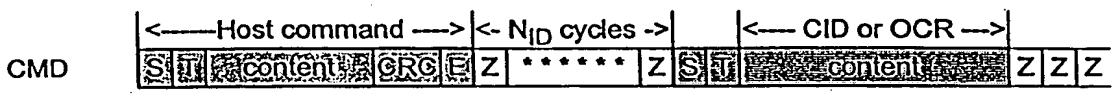



Fig 54

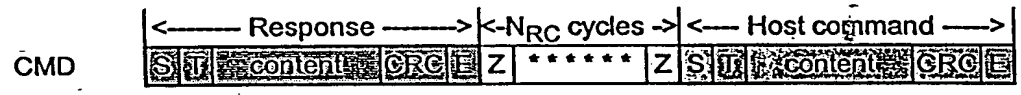



Fig 55

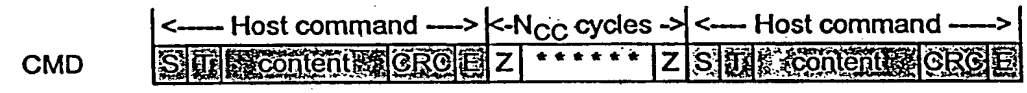



Fig 56

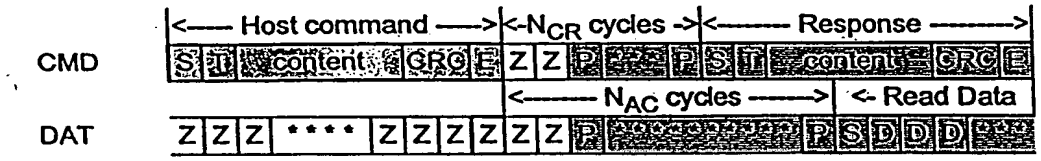



Fig 57

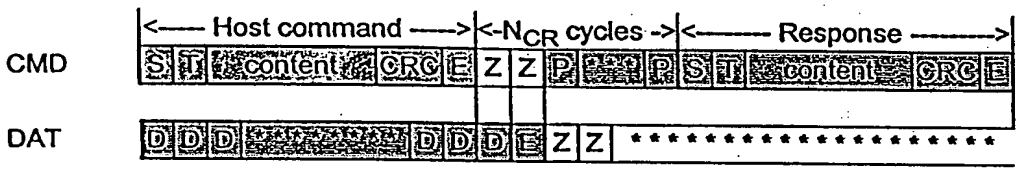



Fig 58

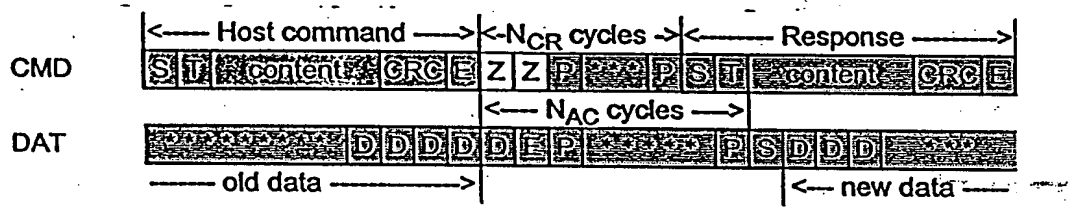



Fig 59

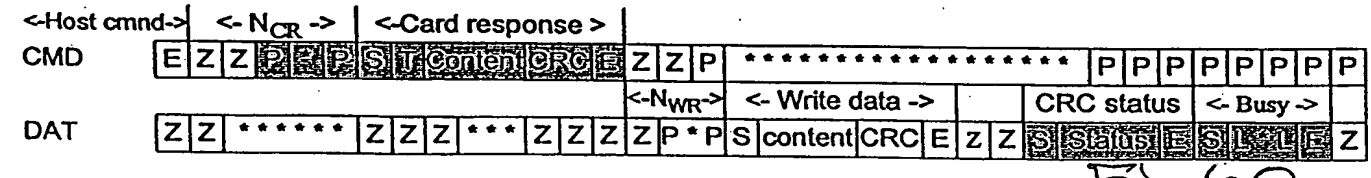



Fig 60

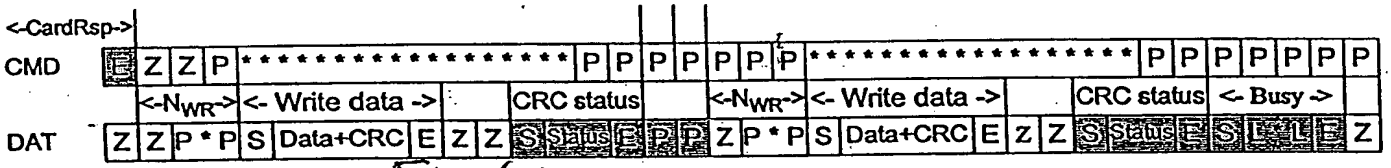



Fig 61

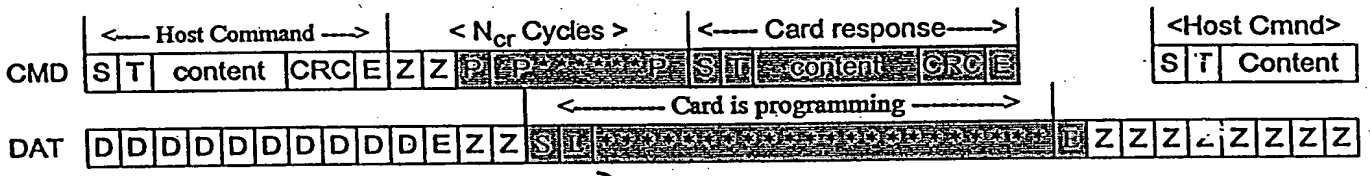
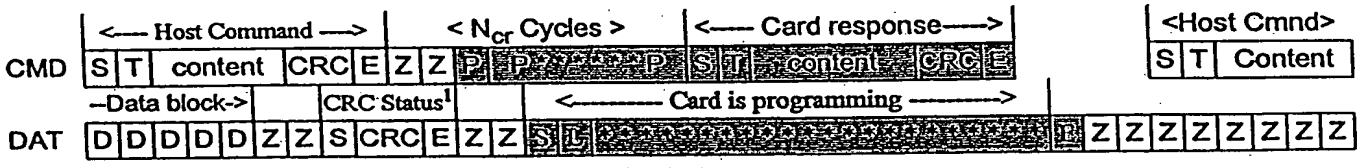



Fig 62




1) The card CRC status response was interrupted by the host.

Fig 63

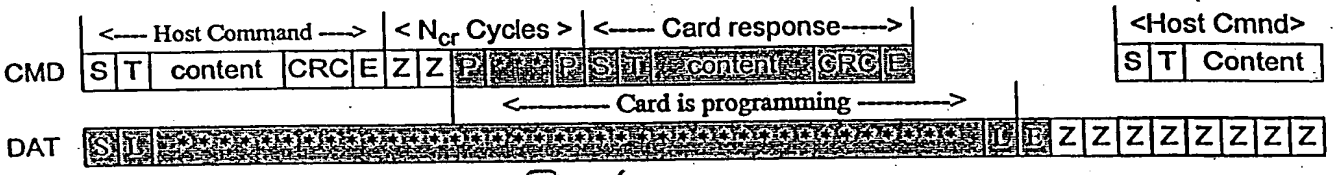

<— Host Command —>
< $N_{cr}$ Cycles >
<— Card response —>
<Host Cmnd>
CMD
S T content CRC E Z Z P P P S T content CRC E
S T Content
<— Card is programming —>
DAT
S L E Z Z Z Z Z Z Z Z

Fig 4

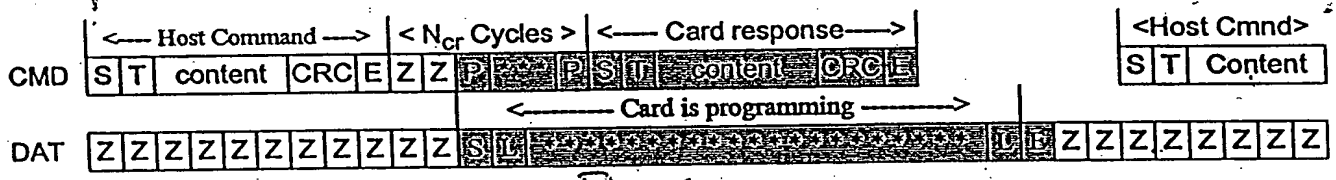

<— Host Command —>
< $N_{cr}$ Cycles >
<— Card response —>
<Host Cmnd>
CMD
S T content CRC E Z Z P P P S T content CRC E
S T Content
<— Card is programming —>
DAT
Z Z Z Z Z Z Z Z Z Z S L E Z Z Z Z Z Z Z Z

Fig 5

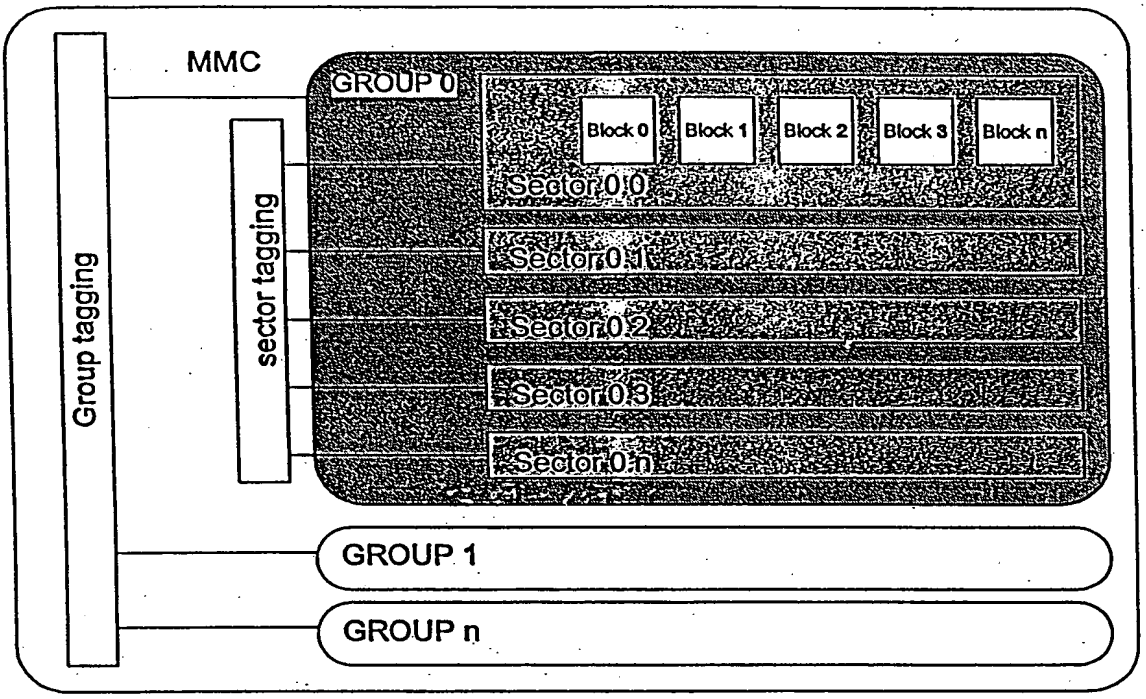

MMC
Group tagging
sector tagging
GROUP 0
Block 0
Block 1
Block 2
Block 3
Block n
Sector 0 0
Sector 0 1
Sector 0 2
Sector 0 3
Sector 0 n
GROUP 1
GROUP n

Fig 66

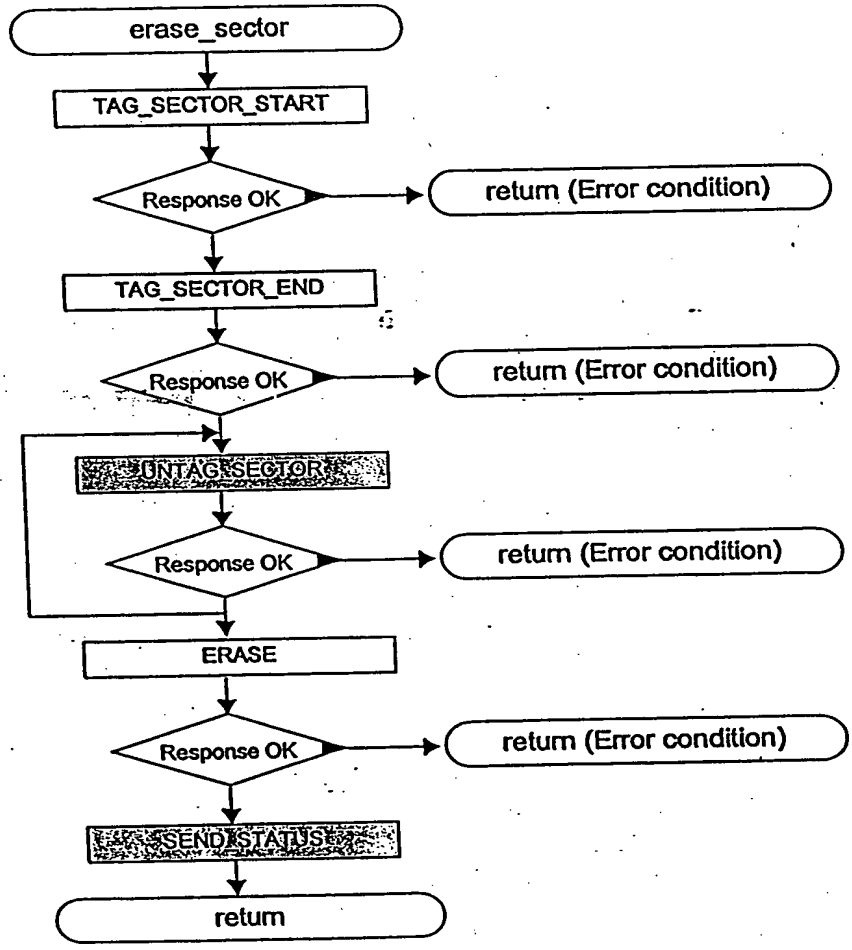

erase_sector
TAG_SECTOR_START
Response OK
return (Error condition)
TAG_SECTOR_END
Response OK
return (Error condition)
UNTAG_SECTOR
Response OK
return (Error condition)
ERASE
Response OK
return (Error condition)
SEND_STATUS
return

Fig 67

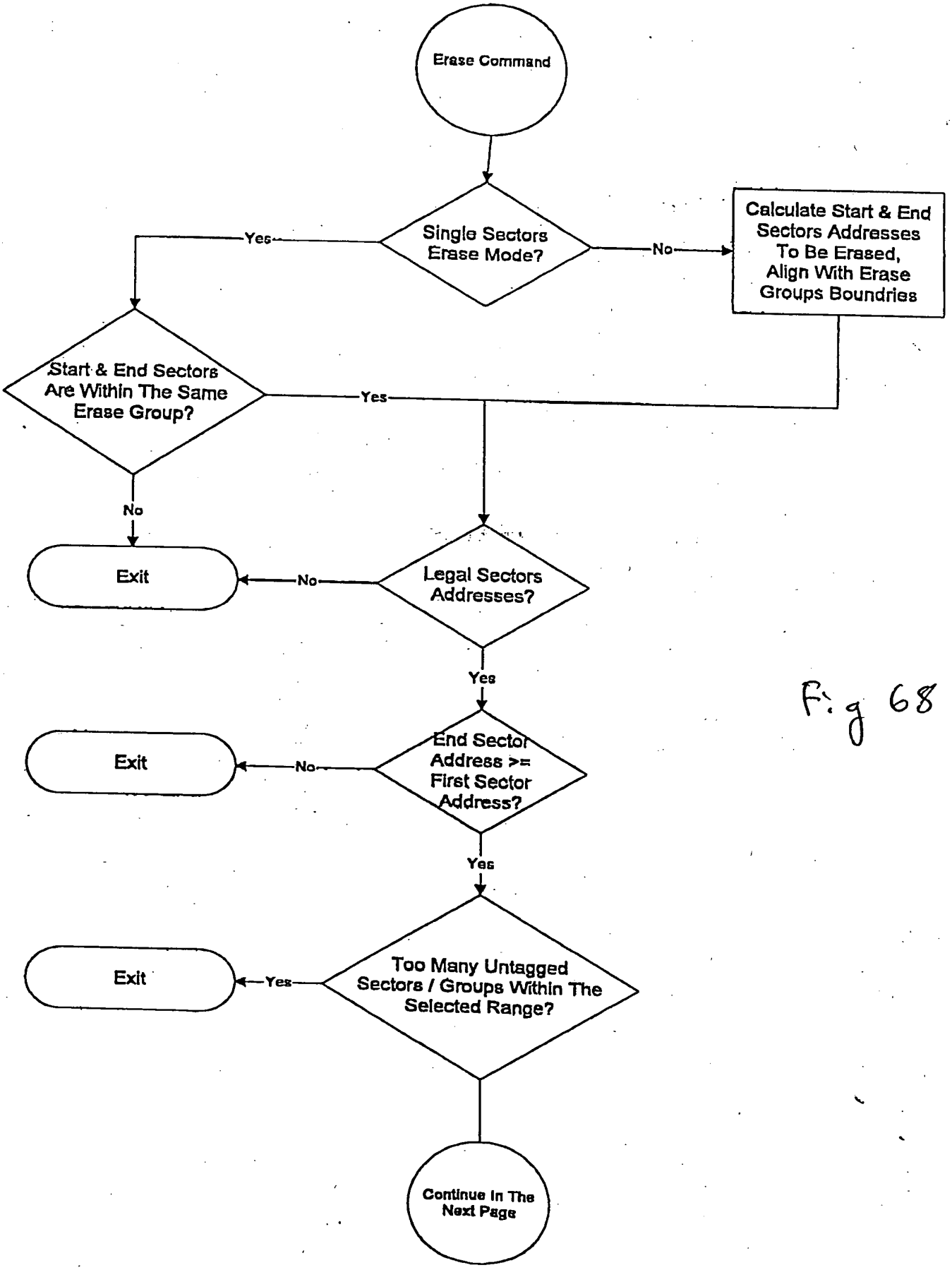

Erase Command
Single Sectors Erase Mode?
Yes
No
Calculate Start & End Sectors Addresses To Be Erased, Align With Erase Groups Boundries
Start & End Sectors Are Within The Same Erase Group?
Yes
No
Exit
No
Legal Sectors Addresses?
Yes
Exit
No
End Sector Address >= First Sector Address?
Yes
Exit
Yes
Too Many Untagged Sectors / Groups Within The Selected Range?
Continue In The Next Page

Fig 68. a

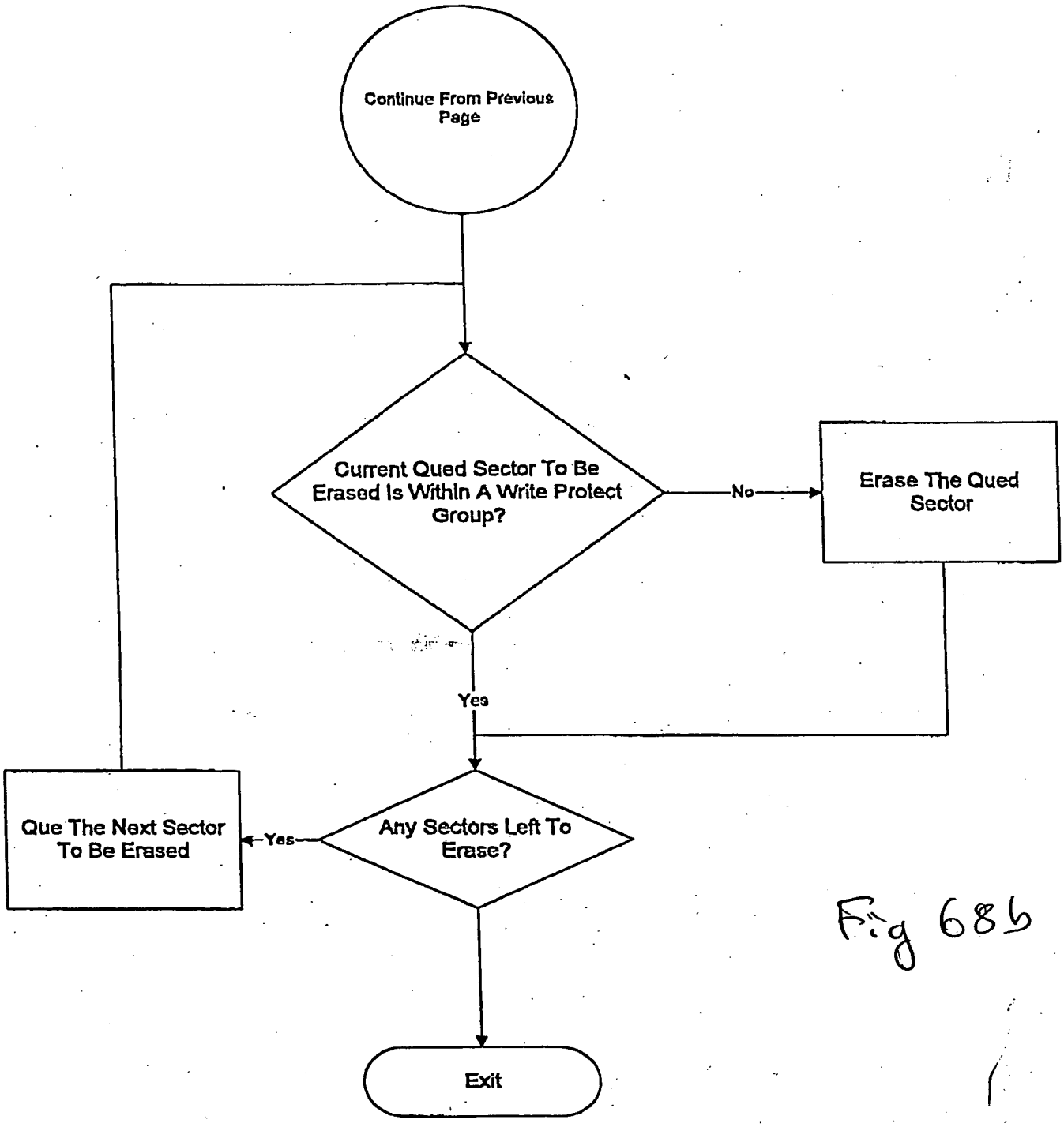

Continue From Previous Page
Current Qued Sector To Be Erased Is Within A Write Protect Group?
No
Erase The Qued Sector
Yes
Any Sectors Left To Erase?
Yes
Que The Next Sector To Be Erased
Exit

Fig 68b

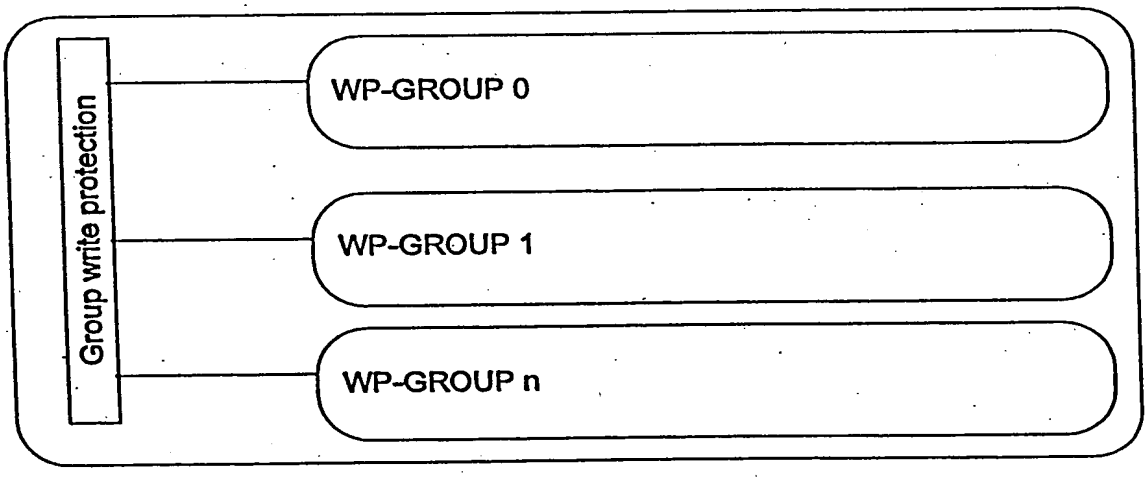



Fig 69

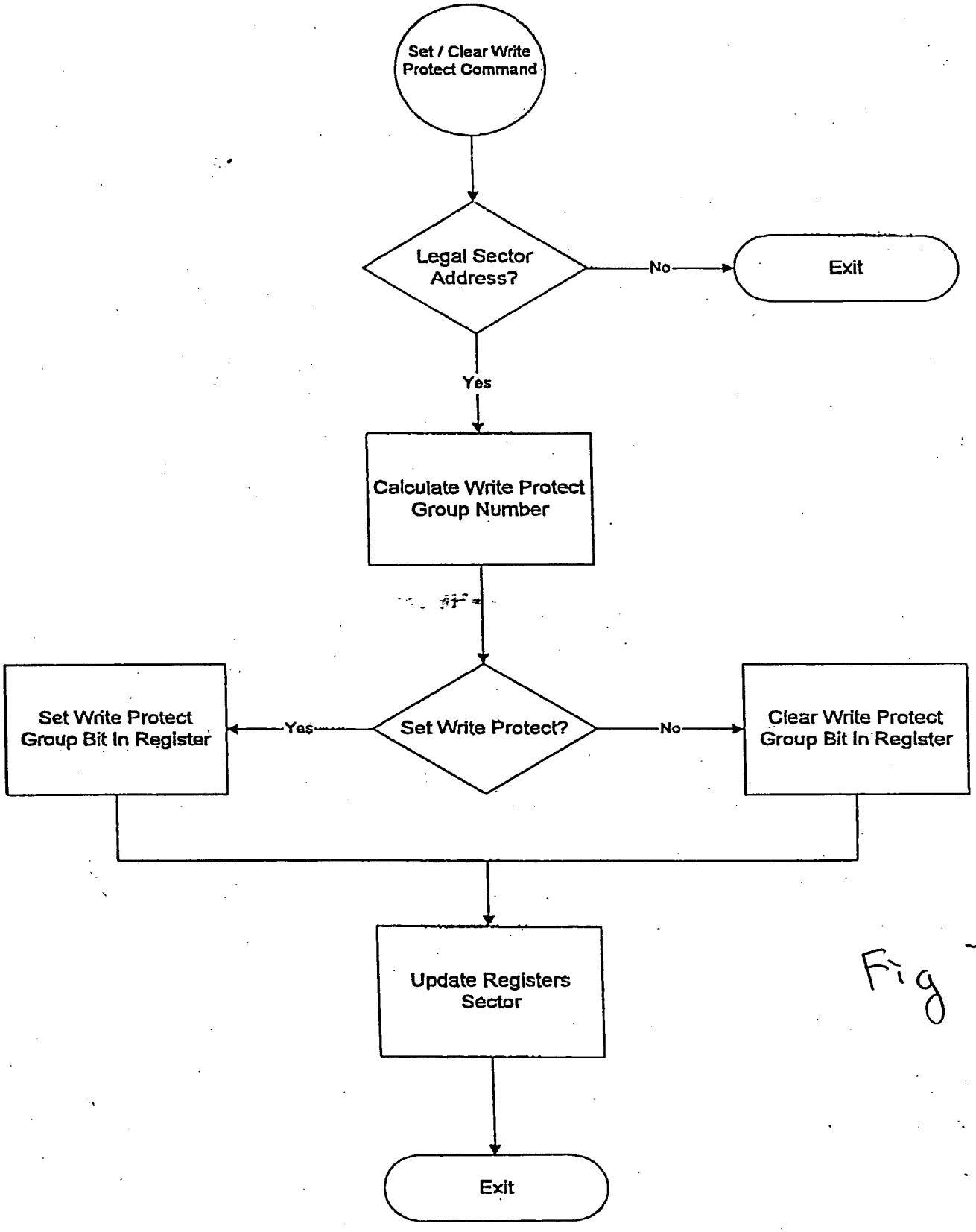

Set / Clear Write Protect Command
Legal Sector Address?
No
Exit
Yes
Calculate Write Protect Group Number
Set Write Protect?
Yes
Set Write Protect Group Bit In Register
No
Clear Write Protect Group Bit In Register
Update Registers Sector
Exit

Fig 70

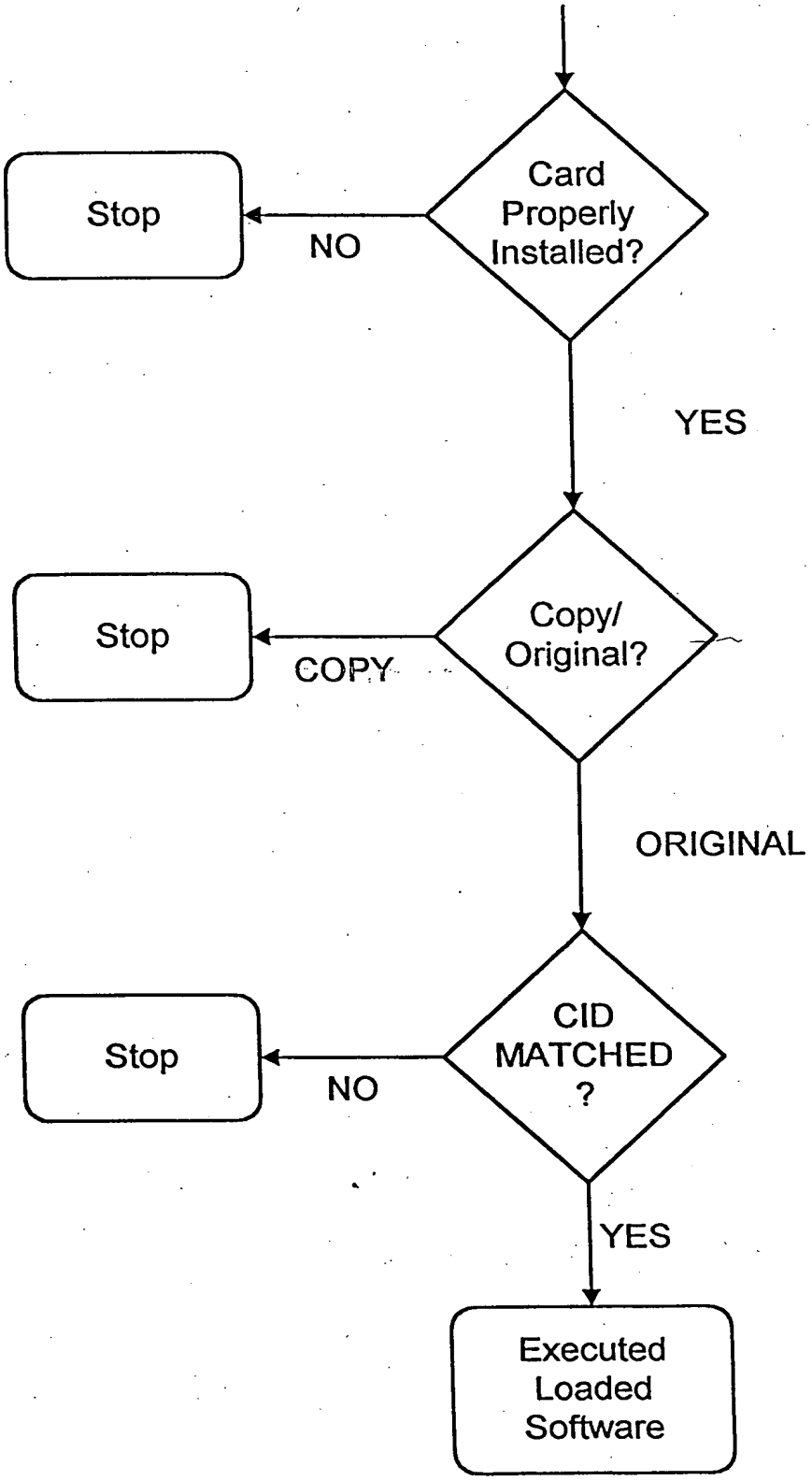

Card Properly Installed?
NO
Stop
YES
Copy/ Original?
COPY
Stop
ORIGINAL
CID MATCHED ?
NO
Stop
YES
Executed Loaded Software

Fig 71